Chapter.02 데이터 분석 라이브러리-04. Pandas를 사용하는 이유

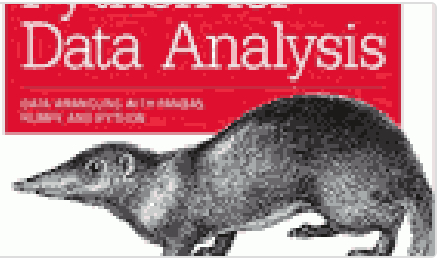
Pandas : Python Data Analysis Library. 정형 데이터 분석에 최적화된 라이브러리.

Pandas!

pandas

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

https://pandas.pydata.org/



- 2008년에 만들어졌으며, 2009년에 100% 오픈소스가 되었습니다.
- 정형 데이터를 효율적으로 표현할 수 있는 `DataFrame` 형태로 모든 데이터를 표현합니다.
  - 다양한 데이터 조작 기능을 제공합니다.
    e.g. indexing(=search), filtering, reshaping, concatenation, reading/writing, ...
- 벡터 연산에 최적화되어 있습니다. → Numpy와 연관성이 있다!

```
# pandas example
import pandas as pd

df = pd.DataFrame(np.random.randn(5, 3))
df.head()
```

Pandas를 사용해야 하는 이유

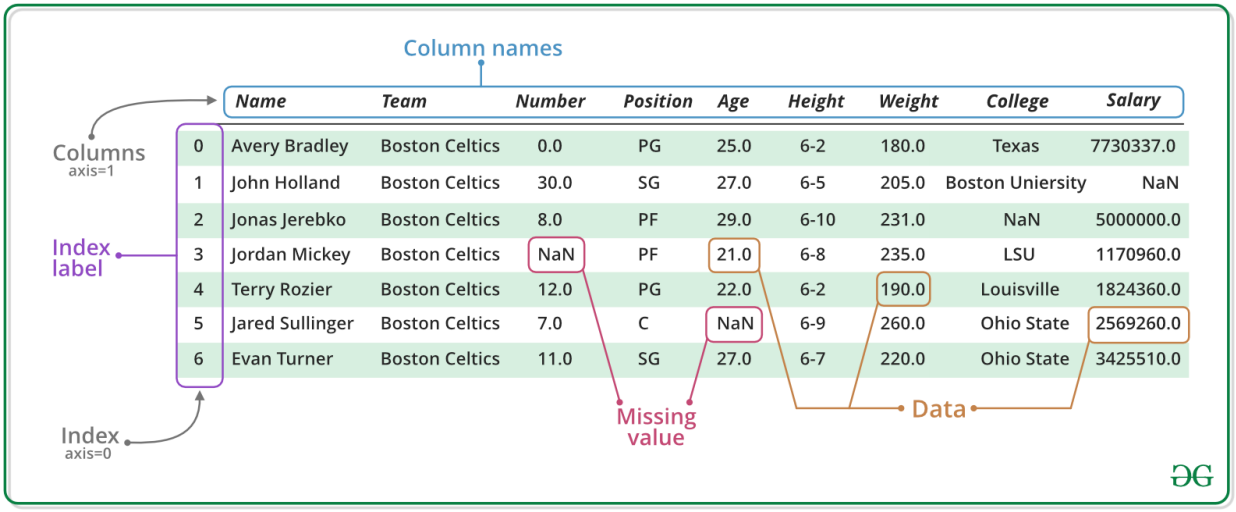
1. 대부분의 정제된 데이터들은 테이블 형태로 표현됩니다. 이런 테이블 형태의 데이터를 분석하기에 최적의 라이브러리입니다.
2. numpy처럼 정형화된 데이터 연산에 최적화 되어 있습니다. 성능이 매우 뛰어납니다!
3. 다양한 정형 데이터를 통합 관리할 수 있습니다. **json, html, csv, xlsx, hdf5, sql, ...** 모두 DataFrame으로 통일해서 표현될 수 있다.
4. 엑셀에서 제공하는 연산 기능을 거의 다 제공합니다. 편의성이 좋다!

Hands-on

1. 정형 데이터 타입인 json, html, csv, hdf5에 대해서 조사해보세요.
2. 정형 데이터 분석과 벡터 연산이 어떤 관련이 있을지 생각해보세요.

Chapter.02 데이터 분석 라이브러리-05. Pandas DataFrame

💡 Pandas DataFrame : pandas 라이브러리가 사용하는 기본 자료구조.



Source : https://www.geeksforgeeks.org/creating-a-pandas-dataframe/

- DataFrame은 2차원 테이블 구조를 말합니다.
- 1차원 구조인 `Series`도 있습니다. (1 row, 1 column)
- row, column으로 모든 원소를 구분합니다. (indexing)
- index, columns, values라는 객체 변수를 가지고 있습니다.
- Relational DB와 완전히 호환됩니다.
- 하나의 column을 기준으로 모든 원소의 data type이 동일합니다. (모두 numpy array가 가지는 data type과 동일)

- DataFrame은 numpy array를 상위 호환하는 개념으로 universal function이 사용 가능합니다.
  → 내부 구현체로 numpy array를 사용하기 때문에!